Published in IET Software
Received on 16th April 2010
Revised on 19th July 2010
doi: 10.1049/iet-sen.2010.0046

In Special Issue on Social Semantic Web Support for Software Development

ISSN 1751-8806

Usage of social and semantic web technologies to design a searching architecture for software requirement artefacts

J. Chicaiza[1] *J. López*[1] *N. Piedra*[1] *O. Martínez*[2] *E. Tovar*[3]
[1]*Escuela de Ciencias de la Computación, Universidad Técnica Particular de Loja, Loja, Ecuador*
[2]*Centro de Investigación Operativa, Universidad Miguel Hernández, UMH, Alicante, Spain*
[3]*Facultad de Informática, Universidad Politécnica de Madrid, UPM, Madrid, Spain*
E-mail: jachicaiza@utpl.edu.ec

Abstract: At present, the research community recognises a complementary relationship between the semantic and the social web. The merging of these web instances could play an essential role in different knowledge domains. In this study, the authors promote a social–semantic web paradigm using software engineering as the knowledge domain specifically. The authors address a major problem – the difficulty for end-users in finding documentation related to software requirements proposed by them; this fact reduces their participation at the time of specifying the software requirements. Architecture is proposed for enhanced resources search, combining the strengths of the social (social annotations) and semantic (semantic metadata) technologies, which has been designed considering the search style of the information seekers. Such architecture is applied in a use-scenario, where the expert users who are not technicians have some restrictions and limitations to retrieve the documents they need. The preliminary results demonstrate that it is possible to take advantage of the defined infrastructure of the ontology to organise and integrate the metadata of resources which are in databases or existent files; this approach opens several possibilities as creation and validation of software requirements collaboratively among different expert-users.

1 Introduction

At the beginning of the last decade, Witold [1] refers to a couple of inner characteristics of software engineering (SE), 'is inherently knowledge intensive and software processes and products are human centered'. Today, through technologies and philosophies of the semantic and social web, it is possible to manage these dimensions of SE.

The need for adding semantic technologies in SE is proposed by Garcia-Crespo *et al.* [2], who recognise the need for counting with semantic metadata in the description of the documentation inside a project for software development. The results of this work, as others referred, demonstrate that the application of this kind of technology in SE scope is hopeful. The semantic web provides the key to large-scale data integration [3]. This feature added to the intelligent information processing capability, makes the sharing and re-use of software elements easier.

However, the semantic web 'still lacks approachable interfaces allowing contributions from non-specialists' [3]; hence, the need for applying social web technologies in software projects becomes evident. In this way, regular users can contribute content, generating a 'collaborative and innovating' [4] ecosystem.

One of the SE activities that requires a social–semantic approach is the management of software requirements. From this point of view, the software requirements and specification documents are created in a collaborative way by the project stakeholders. The incorporation of experts and end-users to validate requirements is of extreme importance.

Most of the requirement managing tools offer searching mechanisms, either complex or too basic. Regular users face difficulties when trying to find the resources they are interested in. Locating the required piece of information can still present challenges to the user [3].

In this study, the authors propose architecture for enhanced resources search, combining the strengths of the social and semantic technologies for retrieving information using basically semantic metadata and social annotations. Such architecture is applied in specific use scenario, where the expert users who are not technicians have some restrictions and limitations to retrieve the documents they need.

The remainder of this paper is organised as follows. Section 2 analyses the synergy between the social and the semantic web to improve or make SE processes easier and it describes the problem of application of the social–semantic web approach to requirements engineering. Section 3 describes the proposal for the resource search. Section 4 explains the use scenario of the proposed architecture. Finally, Section 5 presents the conclusions.

2 State-of-the-art use of semantic and social technologies in software engineering

One of the reasons for which social web or Web 2.0 became so popular is that it is focused on contents, relations and knowledge and not precisely on technology [5]. The tools and services based on social philosophy have been used by a large number of normal web users and domain experts, to generate their own resources, collaborate on a product development or to tag and classify their web resources and later share with other people.

On the other hand, the semantic web, through its technologies allows one to structure and semantically enrich the content – to define and use common vocabulary, to generate new knowledge or to solve word–meaning problems. These and other capabilities made it possible to improve or automate certain tasks that a human agent would not be able to perform. The semantic web has an inner capability for processing a large amount of information.

The synergy between social and semantic web instances is analysed by Gruber [6]: 'The social web is an ecosystem of participation, where value is created by the aggregation of many individual user contributions. The semantic web is an ecosystem of data, where value is created by the integration of structured data from many sources'. In Section 2.1, certain aspects of this approach and its application in SE are shown in detail.

2.1 Social-semantic technologies driven to software engineering

The relation between semantic web and social web or Web 2.0 is that, 'these two approaches are complementary and that each field can and must draw from the other's strengths' [7].

This cooperative approach is called, social–semantic web (see Fig. 1). Torniai *et al.* [8] mention this merging saying that it will allow, 'creating, managing and sharing information through combining the technologies and approaches from Web 2.0 and the semantic we'. Merging the best of both worlds can play a crucial role in different domains.

On the direction semantic-to-social, the contribution is given through the semantic enrichment of tags or content created by users, through social tools. An example in [9], where semantic data are associated to web pages links, based on domain ontology and the user gives opinions about the link content.

The other point of view is social-to-semantic direction; most of the efforts are focused on the usage of social annotations and folksonomies to create and to populate ontologies.

After an analysis of the application of semantic web technologies in software engineering, probably the more common proposals are the ones that use ontologies [10]. There are some studies about organising the knowledge domain in SE and allowing both tools and developers to share information and work cooperatively. In Falbo *et al.* research [11], this problem is put forward into the semantic software engineering environment (SEEs) and their approach is named ODE (ontology-based software development environment).

The re-use of knowledge is presented by Gómez-Berbís *et al.* [12]. They present ESACake 'a semantic software environment for sharing software projects knowledge based on the ESA software methodology', enabling semantic and social interaction of the produced documentation through software development processes. It also uses the project's metadata to optimise the searching process.

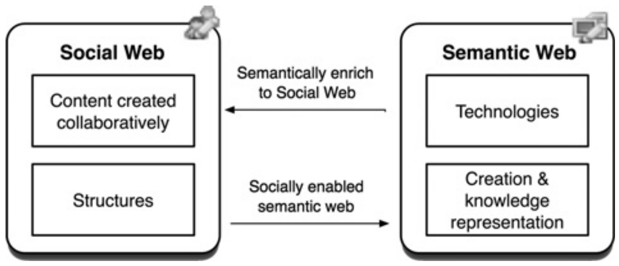

Social Web
Content created collaboratively
Structures
Semantically enrich to Social Web
Socially enabled semantic web
Semantic Web
Technologies
Creation & knowledge representation

Figure 1 *Contributions between social and semantic webs*

Obviously, the enhancement that the semantic technologies would contribute to SE, would be at a product and software processes level. However, to empower the human and social component of the development equipment and support the collaboration of people across organisational boundaries for distributed software engineering [10], it becomes necessary to adopt social web technologies [13].

To extrapolate knowledge and re-use products from the software life cycle, a social semantic network has been proposed [14]. The semantic web is present, adding semantic metadata to a software repository, which makes tasks such as sharing and discovering possible to be done automatically or semi-automatically, whereas the social component is obtained by supplying metadata through a labelled system.

2.2 Problem of application of the social-semantic web approach to requirements engineering

The software projects increasingly grow in complexity and size and require the intervention of experts in different areas. One of the problems associated with this growing is the complexity for managing the software requirements. A problem that has been detected in work teams is the difficulty of end-users in finding documentation related to the software requirements proposed by them. As a consequence, it decreases their participation level and the specification of the software requirements will be incomplete.

Colomo-Palacios *et al.* [14] present two specific applications of the social–semantic technology on the field of requirements engineering – social approach for requirements determination and semantic approach for requirements specification.

Tools for requirement management offer a searching service based on metadata searching. Similarly, these tools in many cases work over a local file repository and the query execution is limited to the enterprise network cover. They require a software installation in client equipment of the users and people typically are unfamiliar with the vocabularies and underlying operations of the systems [15].

Therefore the need for providing easy-to-use and effective tools emerges for searching and also defining a common vocabulary and making it possible to share artefacts and process of software among organisations around the world, to retrieve these artefacts and enable interoperability among different domains.

For searching of software artefacts, the development and usage of vertical search engines (VSEs) could be the option to find more accurate results. However, the usage of specialised searching engines could help to obtain higher accuracy in results. If to the specialised search is added, the semantic component to give structure and semantic to each of the resources and its contents (through metadata standards [16] and ontologies [17]), then, people could solve the problem 'the shortness in understanding user's query intention that occur in traditional search engine' [18].

The most common inclination of semantic searchers such as Truevert (www.truevert.com) or Sensebot (www.sensebot.net) is the usage of automatic methods to identify concepts of an information corpus or to learn language meaning. Another tendency is the use of linguistics to answer specific user questions. The common denominator of these proposals is to incorporate sophisticated components, which demand specialised knowledge from their developers and substantial resources.

Nevertheless, this proposal tries to demonstrate that it is possible to implement a searching engine of acceptable performance, in a simple way and trying to take advantage of the best capabilities of the semantic and social web technologies. In this work, architecture is proposed for resources search, based on an ontology and collective knowledge created by users through social annotations and recommendations.

By means of metadata enrichment and logic inference, people will obtain more precise results from search engines. And at the moment of determining the relationship among the software items (requirements, use cases etc.), social annotations and expert recommendations, the system itself will be in charge of recommending action paths for information seekers.

2.3 Technologies for semantic applications development

The more used technologies (tools and languages) for application development based on OWL ontologies or RDF, are query languages (QL), semantic processing APIs and RDF managers and databases.

QLs are applied to retrieve information from semantic documents. To find out the content of an OWL file, there are languages such as SPARQL-DL [19] and OWL-QL [20]; although both options allow one to explore its capability of expressiveness, their level of implementation and maturity is still incipient.

On the other hand, RDF QLs such as SPARQL, RDQL and SeRQL are more popular and currently used by application developers who incorporate semantic technologies. Of the three already mentioned languages, SPARQL is generally more frequently used for retrieving data, because it is W3C recommended and is supported by recognised databases.

Semantic processing APIs are used to handle objects and data of OWL or RDF resources (either directly from the source files or from databases). Tools such as Jena and Sesame offer querying and inferencing features. In Table 1 the main differences between these two tools are enlisted.

Jena offers three main advantages over Sesame; it is a leading semantic web programmers' toolkit [21]. It also offers direct support to work with SPARQL and it is an easy-to-use API.

About databases, currently, some relational databases offer support for RDF triplets storing. Among the more recognised databases and RDF managers are: 4store, Bigdata, Mulgara, Virtuoso, Sesame and Oracle. Table 2 shows some differences between these technologies: platforms they use to work with, the means of semantic description they support and security they offer.

As seen in Table 2, most of the databases are multi-platform and they support RDF and SPARQL (only Bigdata supports OWL ontologies). Different security mechanisms have been implemented in their infrastructure, this feature is of great importance when performing commercial implementations.

3 Architecture for searching of requirements engineering artefacts

In this section, authors introduce details dealing with the searching architecture for software requirement items. First, it is necessary to describe design principles and criteria that base this architecture supported by ontology, social annotations and recommendations made by experts on software engineering and more accurate requirements engineering.

The architecture is described in Section 3.1, the implementation of two of their components, ontology and query engine, are presented in Sections 3.2 and 3.3, respectively. How it is used will be explained in Section 4; these components have been applied for the search of

Table 1 Jena and sesame comparison

	OWL API	Documentation	Support QL	Supports inferencing	Access to data
Jena	supports for RDFS and OWL	tutorials, mail list and groups	SPARQL	RDF, RDF(S) and OWL	Jena database tables or other relational database
Sesame	no supports	lot of documentation and it may prove to be too difficult for less experienced users	SeRQL	RDF and OWL using an external inference engine	offers in memory, native and remote access to RDF data

Table 2 Comparison between RDF stores and databases

Database name	Platform	Technology	Query language	Security
4store	Unix, Mac OS X	RDF	SPARQL	does not present security risks
Bigdata	Windows, Mac, Linux y Solaris	RDF, RDFS, OWL Lite	SPARQL	strong, fast and manages the concurrency in an efficient and secure way
Mulgara	Windows, UNIX, Linux, Solaris, Mac OS X e IRIX[7]	RDF	ITQL, future support for SPARQL	makes a security copy using Workbench
Virtuoso	Windows, Mac OS X, Linux y UNIX	RDF, XML	ISQL, SPARQL	privileges, roles and hierarchies are managed; also it is included inside the database engine, the data encryption to protect the transmitted data
Sesame	Windows, Unix, Solaris, Mac OS X e IRIX	RDF, RDFS	SeRQL y SPARQL	creation of users and passwords with writing and reading privileges
Oracle	Windows, Linux, Mac OS X, Solaris y HP-UX	RDF, RDFS	SQL and support for embedding SPARQL	authentication methods, secure access to the ports, checklist for the auditory and encryption methods

different requirements artefacts, which are generated in a local software development project.

3.1 Description of the searching architecture

Before describing the components of the architecture, principles and criteria that support the design are described.

When people search on the web, specific situations are frequently found [15, 22].

Principle 1: They do not know how to 'choose the correct words to represent their information problems'.

Principle 2: Generally enter short queries.

Principle 3: Do not usually modify our queries.

Principle 4: Do not usually look at more than the first ten results.

Also, to help common users to improve their search, there are some proposals from the technological point of view. In [23], the link between social and semantic is explained in the following way: 'the ontology metadata provides the benefit of enabling a semantic search engine to find accurate results and to apply reasoning procedures on the metadata'. Respecting the social dimension, Wu *et al.* in [24] state that 'social annotations remove the high barrier to entry because web users can annotate web resources easily and freely; it directly reflects the dynamics of the vocabularies of the users and thus evolves with the users'. Another work that evaluates the social contribution is [25] 'domain-specific sites might have higher quality tags due to the shared context of the users'. Such studies allow us to confirm that

Principle 5: To get an enhanced search, the most common tendency is to integrate social and semantic web technologies.

In order to accomplish these principles, ten criteria that should be met by the architecture have been stated in Table 3.

The result of architecture of the search engine can be seen in Fig. 2. It includes three areas of knowledge management.

1. *Acquisition:* Includes the capture of metadata from any structured data repository (Metadata Reader) and the search of annotations and suggestions made by users on the indexed resources; the component called Annotation Finder would be in charge of obtaining this information, through available APIs in different social services (del.icio.us, twitter etc.).

2. *Encoding and storage:* The metadata obtained in the previous stage, are processed and transformed (Metadata Processor) to the metadata scheme defined by the ontology. Once this process is complete, RDF triplets should be generated to be stored in the database. On the other hand, the ontology that defines the terms vocabulary to describe a resource will be useful to support the inference engine and according to Section 3.2; this has been created using a collaborative process.

3. *Knowledge application:* Two modules will be in charge of exploiting the stored metadata of the triplets store: the query engine makes SPARQL search to retrieve information from the software artefacts and the inference engine will be in charge of dynamically generating a rank of each resource; the score of each resource will be obtained through ranking algorithms, which give priority to the social contribution. (It is not the objective of this work to propose a new ranking algorithm; among the algorithms based on Social Relevancy Rank and would be possible to use, are: FolkRank, SocialPageRank, and SocialSimRank.) A feedback system will allow one to suggest user multiple tags to improve its search attempts (see Table 3, Criteria 6, 7 and 8) or to choose a related resource (see Table 3 Criteria 10).

A future application that implements this architecture should offer a search interface (it is utilised to search educational material), with elements like the ones shown in Fig. 3. Each found resource would be presented tags with which it has been associated and also the related tags (inferred through logic rules). To make a new search, the user would have a possibility of choosing some of the available descriptions.

3.2 Implementation and ontology validation

To semantically represent the metadata of a resource, ontology is used. The creation of a formal model that will be useful to support the software requirement item search, was based on a previous work [26] in which was designed and implemented an ontology to retrieve educational material, OER-CC ontology (http://loxa.ec/semanticweb/ontologies/OER-CC.owl). Using the help of a requirements engineering expert, the own terms of this knowledge domain were identified.

The development of the referred ontology was guided by proposed activities in Methontology [27]. This method proposes an ontology building life cycle based on evolving prototypes. That is, it allows adding, changing and removing terms in each new version (prototype).

Through an iterative and collaborative process, the ontology adaptation was performed. At least two of the requirements [28] that should have a framework for ontology creation are taken into account in this proposal: an easy-to-use and easy-to-communicate graphical representation and automated import/export of OWL and conceptual graphs.

Table 3 Principles and criteria of the architecture

Principle	Criteria number	Architecture feature criteria	Simplicity	Priority to precision rather than recall	Specialised search
5	1	• To use of the metadata described in structured repositories and which offer material related to the interest domain	X	X	X
5	2	• To take advantage of the structure of an ontology, for a better organisation of the information of each resource while database engine manages the performance	X		
4 y 5	3	• For the resulting classification, to exploit the feedback given by common users and experts, instead of incorporating sophisticated or complex ranking algorithms	X	X	
3	4	• To find resources of a determined knowledge domain: to reduce the need for incorporating complex processing components of the natural language or disambiguation	X		X
2	5	• The repository of the search should be constituted by metadata coming from selected information sources: it is not required to execute a crawling of all web resources	X	X	X
1, 2, 3 and 5	6	The search engine should offer the user the following options to help better defining of its information needs: • Each result (resource) will be presented with linked tags and subjects. Next, the user has to be allowed to choose different tags to improve the search		X	
1, 3 and 5	7	• All tags that have been used next to the word or words specified by the user in the search, will be presented in an annotation cloud, among which the user can choose		X	
1, 2, 3 and 5	8	• Users can query with a boolean combination of tags and other keywords		X	
4 and 5	9	• Allows to add annotations to each resource; if a person determines that a resource can be better described, it can be incorporated from the search engine		X	X
1, 2 and 3	10	• All related resources must be presented when chosen by the user: a resource could be associated to other resources through relations of different nature ('it is part of', 'it is version of' etc.)		X	X

Fig. 4 shows the main stages of the ontology creation according to the selected methodology, the output of each phase and the type of user that can participate in a direct or an indirect way. The use of graphic tools as the ones stated (Cmaptools (http://cmap.ihmc.us) and COE Cmaptools (http://coe.ihmc.us/groups/coe/)) has made easier the participation of different users, in most of the activities of the creation. This exposed strategy can be applied for ontology

doi: 10.1049/iet-sen.2010.0046

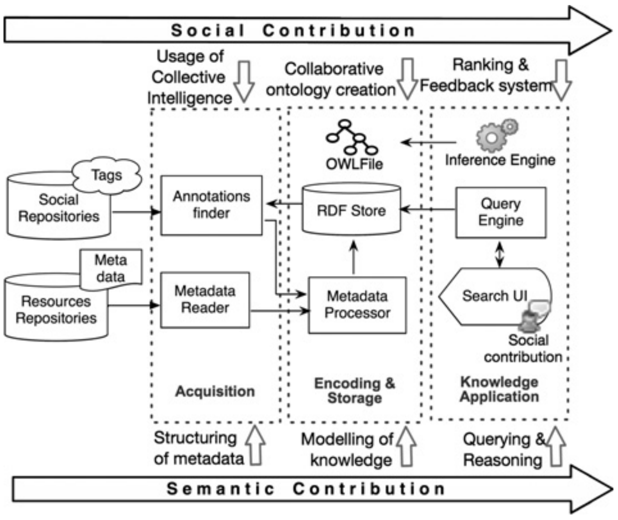
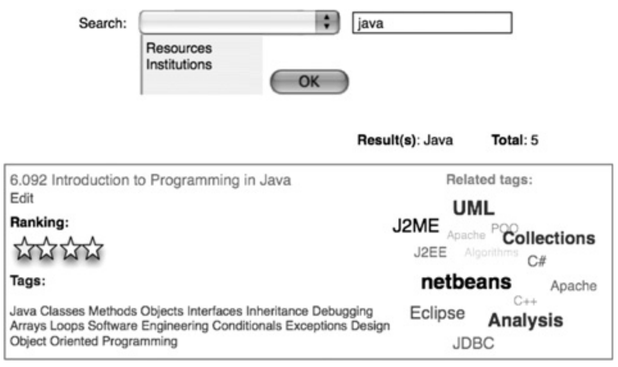


Figure 2 *Logic architecture of the search engine*


Search: [ ] java
Resources
Institutions
OK

Result(s): Java Total: 5

6.092 Introduction to Programming in Java
Edit
Ranking:
☆☆☆☆
Tags:
Java Classes Methods Objects Interfaces Inheritance Debugging Arrays Loops Software Engineering Conditionals Exceptions Design Object Oriented Programming

Related tags:
UML J2ME Apache POO Collections J2EE Algorithms C# netbeans Apache C++ Eclipse Analysis JDBC

Figure 3 *Interface of the search engine*
General results of a query

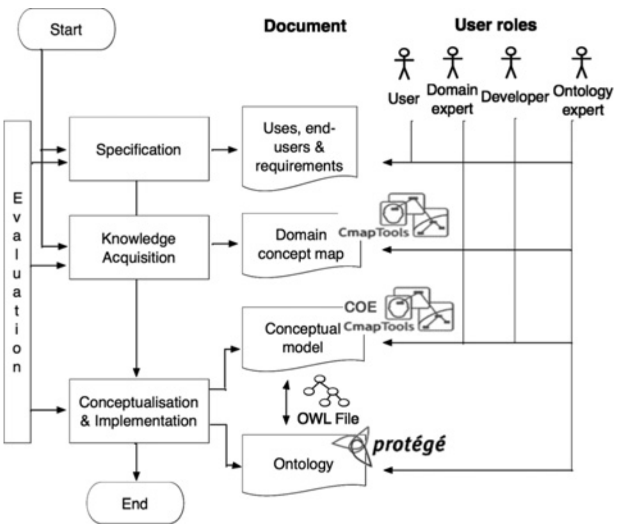


Figure 4 *Process of ontology creation*

management in small- or medium-scale efforts. For the ontology creation of higher scope or that requires greater effort, there are integrated tools such as OntoEdit [29] or visual ontology modeller [28].

During the specification stage, where the ontology requirements are determined, different users can express their need for information. The capture of knowledge from the experts is placed in a graphic model, through the usage of CmapTools.

The necessary efforts to develop conceptualisation and implementation stages of the ontology can be reduced using COE CmapTools. While allowing the creation of a graphic first version of the ontology with the main domain-terms, domains-experts and developers can participate in the creation of this first prototype because this is a legible model for humans. They are capable of providing feedback. Finally, the axioms, rules and the rest of the formalities will be done directly by an ontologist, using the Protégé editor.

In this work, three versions of the ontology were generated, according to the work philosophy proposed by Methontology. Specifically, as shown in Table 4, the ontology was implemented in three stages of progressive improvement.

Additionally, each ontology prototype was syntactically evaluated by means of Pellet 1.5.2 and RacerPro 2.0 reasoners; and a taxonomy evaluation was performed, to verify that there would no mistakes in the taxonomies like inconsistency (see Fig. 5), incompleteness and redundancy of concepts [30].

No errors were detected by the reasoner. Regarding the taxonomy evaluation, no errors were identified.

One of the ways to demonstrate the potential of ontologies to represent knowledge is defining and executing queries and inference rules to retrieve and to exploit its knowledge. Different tools and methods that could be applied for ontology population are referred in [27].

3.3 Search prototype implementation

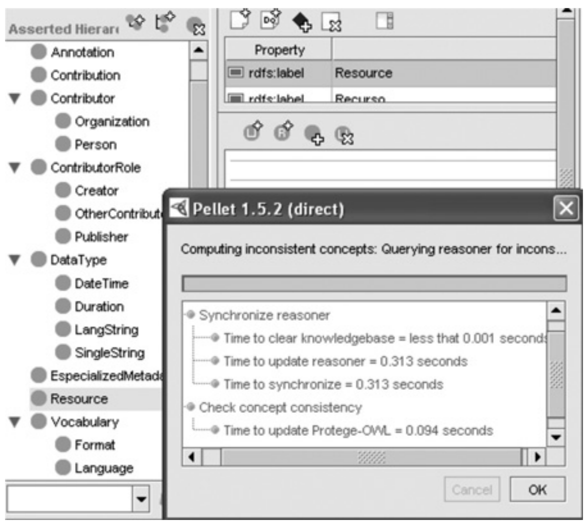
The search prototype has been developed with a layered architecture using an evolving development cycle, based on prototypes. The three layers implemented can be seen in Fig. 6. The semantic component is represented by the OWL file and RDF store, which are used to codify knowledge in the requirements domain and are available on a publicly accessible server.

Fig. 6 also shows the technologies used in the development:

- For the codification and retrieving of knowledge, languages recommended by the W3C, like OWL, RDF and SPARQL were used.

Table 4 Versions of Ontology to support requirements querying

Ontology version	Description	Tool
1.0	considering a subset of metadata for software description items, it has defined the ontology objects of the existent ontology that could be re-used, created and adapted	COE Cmaptools
2.0	this version included the definition of topics such as • transitive, reflexive, functional and has value properties and special classes like disjoint • Enumerations of certain data properties • cardinality restrictions • annotation properties (synonym, acronym, rdf:label and rdfs:comment) for each object of the ontology • new classes for grouping related concepts	Protégé 3.3.1
3.0	this version basically incorporated small changes, with the objective of giving answer (through SPARQL queries) to the questions that are expected to be answered by the ontology	Protégé 3.3.1



Property	
rdfs:label	Resource
rdfs:label	Recurso

```
Computing inconsistent concepts: Querying reasoner for incons...

Synchronize reasoner
    Time to clear knowledgebase = less that 0.001 seconds
    Time to update reasoner = 0.313 seconds
    Time to synchronize = 0.313 seconds
Check concept consistency
    Time to update Protege-OWL = 0.094 seconds
```

Figure 5 *Check consistence of ontology*

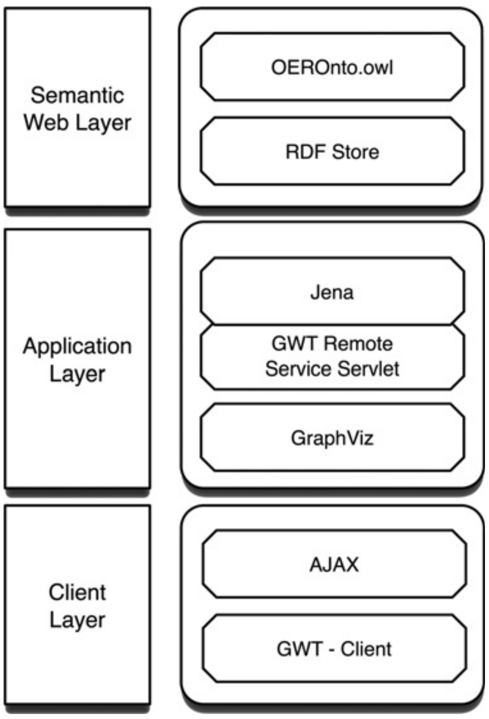


Figure 6 *Application layers of the search engine prototype*

- For the storing of the triplets, the Oracle 11g database was used, which has been selected for its capability of storing a large amount of information.

- For programming Java language was used, which is platform independent.

- Jena APIs have been used for extracting data, OWL ontology for making knowledge inferences and framing queries using SPARQL on RDF Store. Jena is the most used environment to work with ontologies and applications based on RDF.

- GWT supports development of AJAX-enabled applications. The user interface was developed with the support of this tool.

- GraphViz was used to draw RDF graphs of the query results.

- AJAX was used for asynchronous communications with the server and to improve the user's experience.

- Additionally, other APIs were utilised to perform tasks such as formatting the RDF results.

Once the search prototype (http://200.0.28.13:8080/oersearch/) has been developed, the following features stand out – searches for keywords (simple and advanced), provides options for results filtering, its design is simple and similar to the ones offered by other search engines, it also provides descriptions in many languages (if multilingual descriptions of metadata were specified) and the presentation of results is made in HTML and RDF formats (this makes communication with other agents easier and opens the possibility of a future where it will be able to develop a search widget to be added to any web portal).

To check the support of functionalities already mentioned, a concept proof is described in the following section.

4 Use scenario

A software development team from a local institution constitutes the environment in which the model was validated. The biggest development project on which the team is working is an academic management system for the recognised University of Ecuador which has more than 80 university centers distributed nationally and internationally. Approximately 26 000 users such as students, faculty and administrative and financial staff will use the system.

During requirement specifications, different end users must participate in this process, among others such as secretaries, accountants, teachers and managers. The workflow in which the team manages to state the user's requirements is – a user asks for a requirement; the software analyst writes the specification in a document, catalogues this resource and stores it in a repository; next, a business expert should seek the resource to make it possible to validate and report issues by e-mail to the analyst in charge of the requirement.

Fig. 7 shows the project structure. The application development is divided into several modules, and the unit of the final work are the requirements.

One of the problems identified in the requirement management process is that each analyst incorporates his/her own vocabulary to describe the requirements artefacts; this fact makes it difficult for other people to find them. Also, the used requirement management tool offers very limited functionality for searching and the business expert or final user must explore many incorrect results, before finding the expected document and it requires to be installed before being used.

The study case and the difficulties experienced by the users when they want to find their requirement artefacts were exposed, next, the activities performed to test the proposed architecture are presented.

The metadata of each resource, are the ones generated in the development project of the academic management system. Currently, more than a thousand requirements and nearly 300 use cases are registered.

To map the set of original metadata to the defined structure by the ontology, a model of the intermediate data shown in Fig. 8 was used. Each entity of the structure in Fig. 7 (project, model, use-case and requirement) is considered as a resource.

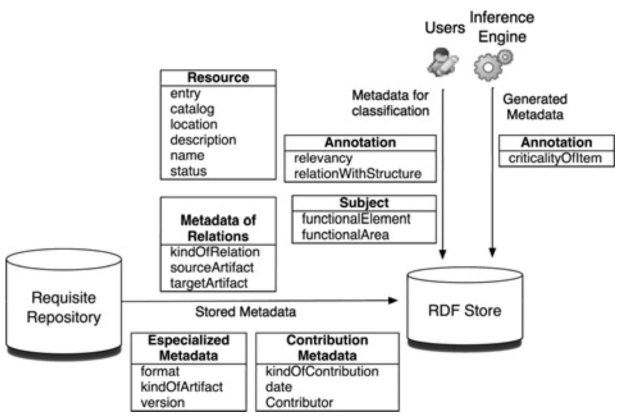


Figure 8 *Intermediate data model*

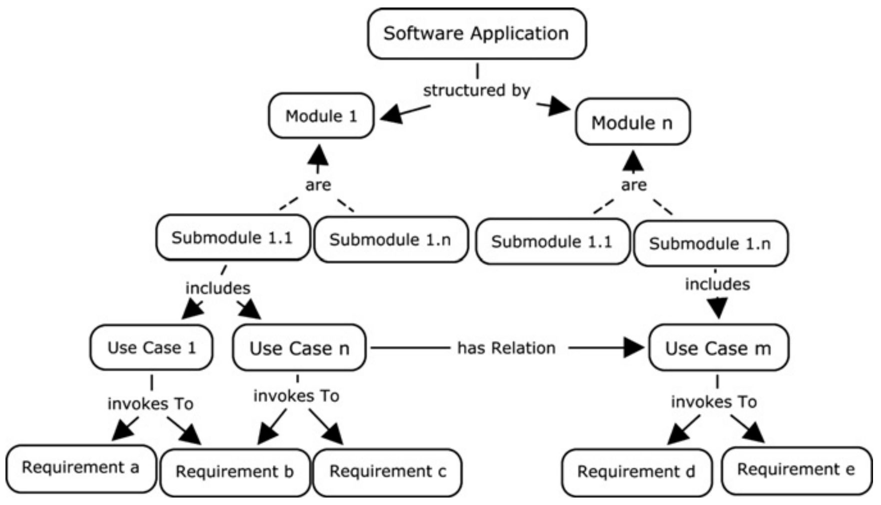


Figure 7 *Structure of software application – study case*

Different categories of metadata were found in the repository – technicians described by the software analysts; subjects related to the functional area in which are enclosed a specific item, registered by the business experts and status metadata generated through the execution of inference rules.

Once the data have been processed and structured according to the described model, a program was implemented to process metadata, to generate RDF triplets and to store them in the database used.

From this moment, one of the main beneficiaries of the system, a business-expert is able to make queries through the web interface of the search engine. Before he starts interacting with the system, a preliminary explanation is given.

When the user's behaviour with the system was observed, the following facts were detected – immediate familiarity with the interface and with the query mechanism, the use of one or two keywords, immediate identification of the resource required due to the multiple tags and other metadata that are shown for each resource and the options of advanced search were just used when the required resource was not located in the first page of the results.

Once the user has explored the system, the following impressions were mentioned – the search mechanism is simple, it facilitates mobility (which is very advantageous, because it is necessary to travel through different parts of the country where the university has its offices) and the time of response is considered to be acceptable enough.

Finally, in scenarios as the one just described, offering subscription service RSS can be very productive for users, because the ones who are specialised in an area will be able to register to receive notifications when new categorised resources are acquired in the area of his functional interest.

5 Conclusions

The main contribution of this work is an architecture for the enhanced search of resources, which tries to integrate in a better way the social (collective intelligence) and the semantic (integration and structure of data) web. When implementing this proposal, a normal user will be able to express in a simple way his information needs and find resources. Hence, the architecture has been designed considering the style of the search of the users.

Having semantically described resources and offering the results of a search in RDF format, the authors think it will contribute to create a web of data, in this way. Different human and software agents will be able to communicate among themselves. Also they will be able to enrich and consume that information for different purposes. The already described use scenario is open to several possibilities when adopting this infrastructure, as the collaborative creation and validation of software requirements, taking advantage of explicit and tacit knowledge about experts in different areas and with experience in similar studies.

The search prototype of software requirement documents generated in a development project has demonstrated that it is possible to take advantage of the infrastructure to organise and integrate the metadata of resources which are in databases or existent files (such as the feeds RSS offered in different websites).

Currently, authors are working to implement other supporting components, such as the implementation of a public query API using the Web REST service that allows developers to integrate the search services into other applications. A social feedback manager is meant to consume annotations and recommendations of users from different services and ranking algorithms based on the descriptors gotten through APIs of social tools.

Finally, to measure the impact of this proposal, the authors are designing a metric system that allows quantitative evaluation and expands the preliminary quantitative results presented in this paper.

6 Acknowledgment

This work has been partially funded by the Madrid Regional Government eMadrid (Ref: S2009/TIC-1650) 'Research and Development for e-learning technologies' and scholarship provided by the Secretaría Nacional de Ciencia y Tecnología of Ecuador (SENACYT).

7 References

[1] WITOLD P.: 'Computational intelligence as an emerging paradigm of software engineering'. Proc. Int. Conf. Software Engineering and Knowledge Engineering, Ischia, Italy, 2002, pp. 7–14

[2] GARCÍA-CRESPO A., COLOMO-PALACIOS R., GÓMEZ-BERBÍS J.M., MENCKE M.: 'BMR: benchmarking metrics recommender for personnel issues in software development projects', *Res. Comput. Sci. J.*, 2009, **2**, (3), pp. 257–267

[3] HEATH T.: 'Information-seeking on the web with trusted social networks: from theory to systems'. PhD thesis, The Open University, January 2008

[4] O'SULLIVAN D., DOOLEY L.: 'Collaborative innovation for the management of information technology resources', *Int. J. Hum. Capital Inf. Technol. Prof.*, 2010, **1**, (1), pp. 16–30

[5] PIEDRA N., CHICAIZA J., LÓPEZ J., TOVAR E., MARTÍNEZ O.: 'Open educational practices and resources based on social software, UTPL experience'. Proc. Int. Conf. European

American Conf. on Telematics and Information Systems, 2009, pp. 1–8

[6] GRUBER T.: 'Collective knowledge systems: where the social web meets the semantic web', *J. Web Semantics*, 2008, **6**, (1), pp. 4–13

[7] ANKOLEKAR A., KRÖTZSCH M., TRAN T., VRANDECIC D.: 'The two cultures: mashing up web 2.0 and the semantic seb', *J. Web Semantic*, 2008, **6**, (1), pp. 70–75

[8] TORNIAI C., JOVANOVIC J., GAŠEVIC D., BATEMAN S., HATALA M.: 'E-learning meets the social semantic web'. Proc. Eighth IEEE Int. Conf. on Advanced Learning Technologies, Washington, DC, USA, 2008, pp. 389–393

[9] GARCÍA-CRESPO A., COLOMO-PALACIOS R., GÓMEZ-BERBÍS J.M., GARCÍA-SÁNCHEZ F.: 'SOLAR: social link advanced recommendation system', *Future Gener. Comput. Syst. J.*, 2010, **26**, (3), pp. 374–380

[10] GALL H., REIF G.: 'Semantic web technologies in software engineering'. Proc. Int. Conf. on Software Engineering, 2008, Leipzig-Germany

[11] FALBO R., GUIZZARDI G., NATALI A., BERTOLLO G., RUY F., MIAN P.: 'Towards semantic software engineering environments'. Proc. Int. Conf. on Software Engineering and Knowledge Engineering, 2002, pp. 477–478

[12] GÓMEZ-BERBÍS J.M., MENCKE M., CHAMIZO J., COLOMO-PALACIOS R., GARCÍA-CRESPO A.: 'EsaCake: a semantic software environment for sharing software projects knowledge based on the ESA software methodology'. Conf. Paper in Proc.: 2008 Third Int. Conf. on Internet and Web Applications and Services, 2008, pp. 535–540

[13] AHMADI N., JAZAYERI M., LELLI F., NESIC S.: 'A survey of social software engineering'. Proc. Int. Automated Software Engineering – Workshops, 2008, pp. 1–12

[14] COLOMO-PALACIOS R., GÓMEZ-BERBÍS J.M., GARCÍA-CRESPO A., PUEBLA-SÁNCHEZ I.: 'Social global repository: using semantics and social web in software projects', *Int. J. Knowl. Learn.*, 2008, **4**, (5), pp. 452–464

[15] BELKIN N.: 'Helping people find what they don't know', *Commun. ACM*, 2000, **43**, (8), pp. 58–61

[16] AL-KHALIFA H.S., DAVIS H.C.: 'Replacing the monolithic LOM: A folksonomic approach'. IEEE Int. Conf. on Advanced Learning Technologies, 2007, pp. 665–669

[17] GRUBER T.: 'A translation approach to portable ontology specifications', *Int. J. Hum. Comput. Stud.*, 1993, **5**, (2), pp. 199–220

[18] ZHI-QIANG D., JING H., HONG-XIA Y., JIN-ZHU H.: 'The research of the semantic search engine based on the ontology', *Wirel. Commun., Network. Mob. Comput.*, 2007, pp. 5403–5406

[19] SIRIN E., PARSIA B.: 'SPARQL-DL- SPARQL Query for OWL-DL'. Proc. Int. Third OWL Experiences and Directions Workshop, 2007

[20] FIKES R., HAYES P., HORROCKS I.: 'OWL-QL. A language for deductive query answering on the semantic web', *J. Web Semantics*, 2004, **2**, (1), pp. 19–29

[21] MCBRIDE B.: 'Jena: a semantic web toolkit', *IEEE Internet Comput.*, 2002, **6**, pp. 55–59

[22] MORRISON M.: 'Tagging and searching: search retrieval effectiveness of folksonomies on the World Wide Web', *Inf. Process. Manage. J.*, 2008, **44**, (4), pp. 1562–1579

[23] HERZOG C., LUGER M., HERZOG M.: 'Combining social and semantic metadata for search in a document repository'. Proc. European Semantic Web Conf., Innsbruck, Austria, 2007, pp. 14–21

[24] WU X., ZHANG L., WU X.: 'Exploring social annotations for the semantic web'. Proc. Fifteenth Int. Conf. on World Wide Web, 2006, pp. 417–426

[25] HEYMANN P., KOUTRIKA G., GARCIA-MOLINA H.: 'Can social bookmarking improve web search?'. Proc. Int. Conf. on Web Search and Web Data Mining, 2008, pp. 195–206

[26] PIEDRA N., CHICAIZA J., LÓPEZ J., MARTÍNEZ O., TOVAR M.: 'An approach for description of Open Educational Resources based on semantic technologies'. Education Engineering (EDUCON), 2010, pp. 1111–1119

[27] GÓMEZ-PÉREZ A., FERNÁNDEZ-LÓPEZ M., CORCHO O.: 'Ontological engineering' (Springer-Verlag, 2002)

[28] CECCARONI L., KENDALL E.: 'A graphical environment for ontology development'. Proc. Second Int. Joint Conf. on Autonomous Agents and Multiagent Systems, 2003, pp. 958–959

[29] SURE Y., ERDMANN M., ANGELE J., STAAB S., STUDER R., WENKE D.: 'OntoEdit: Collaborative Ontology Development for the Semantic Web'. Proc. First Int. Semantic Web Conf. on the Semantic Web, 2002, pp. 221–235

[30] GÓMEZ-PÉREZ A., SUÁREZ-FIGUEROA M.C.: 'Results of axonomic valuation of RDF(S) and DAML + OIL Ontologies using RDF(S) and DAML + OIL'. CEUR Workshop Proc., 2003, vol. 87, pp. 13–26